Encryption Tool - Hacker Style

This program provides a graphical interface for encrypting and decrypting messages using RSA cryptography. Built with Python and `tkinter`, it offers functionalities like RSA key generation, encryption, and decryption in a hacker-inspired black-and-green aesthetic.

Features

1. **RSA Key Pair Generation**:
    a. Generate a private/public key pair.
    b. Save keys in PEM format.
2. **Message Encryption**:
    a. Encrypt plaintext using a public key.
    b. Display encrypted messages in a user-friendly interface.
3. **Message Decryption**:
    a. Decrypt messages using the corresponding private key.
    b. View the original plaintext.
4. **GUI Features**:
    a. A visually engaging hacker-style interface.
    b. Right-click menus for copy/paste actions in all text fields.
    c. Clipboard functionality for copying encrypted text.

How to Use

Requirements

Ensure the following are installed on your system:

- Python 3.8 or higher
- Required Python packages (install via `pip`):

```
bash
CopyEdit
```

```
pip install cryptography
```

Running the Program

1. Save the program code to a file named `encryption_tool.py`.
2. Run the file:

```
bash
CopyEdit
python encryption_tool.py
```

User Guide

1. **Generate Keys**:
   a. Click the **Generate Keys** button.
   b. Save the private and public keys to secure locations.
2. **Encrypt a Message**:
   a. Type the plaintext message in the "Enter Message" box.
   b. Click **Encrypt** and select the public key file.
   c. The encrypted message will appear in the "Encrypted Message" box.
   d. To copy the encrypted text, click **Select and Copy Encrypted Text**.
3. **Decrypt a Message**:
   a. Paste the encrypted message in the "Encrypted Message" box.
   b. Click **Decrypt** and select the private key file.
   c. The original plaintext will appear in the "Decrypted Message" box.

File Descriptions

- `encryption_tool.py`: The main program file.

Known Issues and Limitations

- **Key Security**: Always store private keys securely to prevent unauthorized access.
- **Plaintext Size**: RSA encryption is limited to small amounts of data. For large files, consider hybrid encryption techniques.

- **No Password Protection**: Private key loading does not support password protection.

License

This program is licensed under the MIT License.

MIT License

Contribution

Feel free to submit pull requests or issues if you encounter bugs or have ideas for improvements.

Acknowledgments

This program uses the following libraries:

- `cryptography`
- Python's built-in `tkinter` for the graphical interface.

For questions or support, please contact d.ynacay326@gmail.com